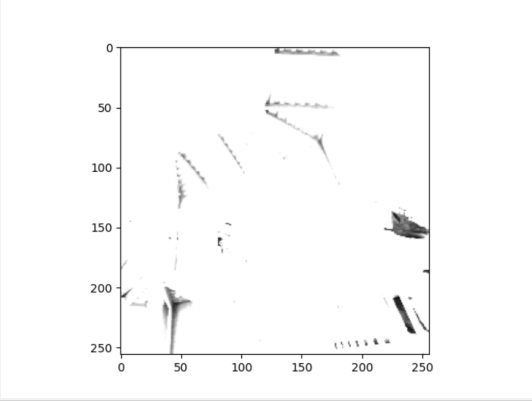
- **Last time you did an exercise (convolutions and pooling) where you manually applied a 3x3 array as a filter to an image of two people ascending an outdoor staircase. Modify the existing filter and if needed the associated weight in order to apply your new filters to the image 3 times. Plot each result, upload them to your response, and describe how each filter transformed the existing image as it convolved through the original array and reduced the object size.**

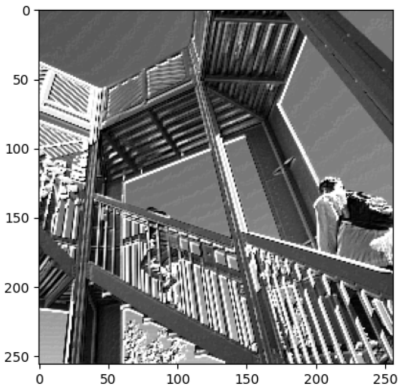
  Filter 1: `[ [-1, 2, 0], [1, 1, 2], [0, 4, 3]]`

  This filter only keeps the darkest spots in the photo and erases the parts that are not dark enough.This filter is in fact quite useless as it does not accentuate or emphasize helpful features.



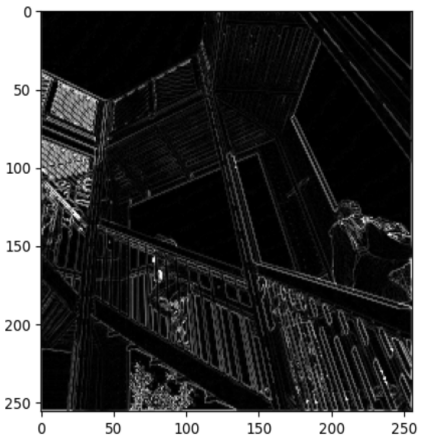
  Filter 2: `[ [-2, -1, 0], [-1, 1, 1], [0, 1, 2]]`

  This filter adds highlight to the image and makes all lines bolder.



Filter 3: `[ [0, 1, 0], [1, -4, 1], [0, 1, 0]]`

This filter completely darkens everything other than the lines that delineate the objects in the image. This filter can be useful to line and edge related detection, even though I am having a hard time to think of a specific example.



- **What are you functionally accomplishing as you apply the filter to your original array (see the following snippet for reference)?**

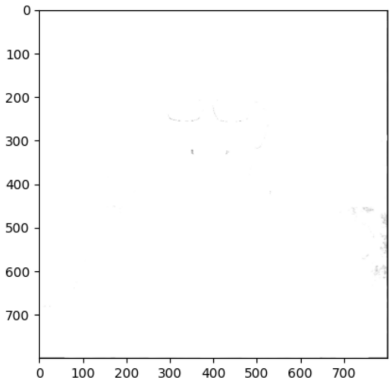
  Mathematically speaking, a linear transformation is taking place in the matrix that represents the original image. After the original matrix is multiplied by the filter matrix, a new matrix with modified value which represents a modified image is produced.

- **Why is the application of a convolving filter to an image useful for computer vision?**

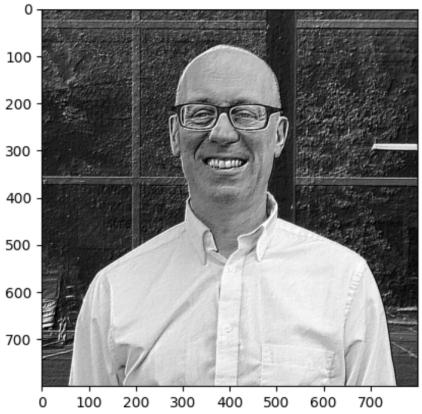
  Sometimes instead of searching information in raw pixels, we want to look at the specific patterns that the image presents in order to categorize the image. The convolving filters can help us to highlight the features that we want to see and help the computer to identify objects more easily.

- **Stretch goal: instead of using the misc.ascent() image from scipy, can you apply three filters and weights to your own selected image? Again describe the results.**

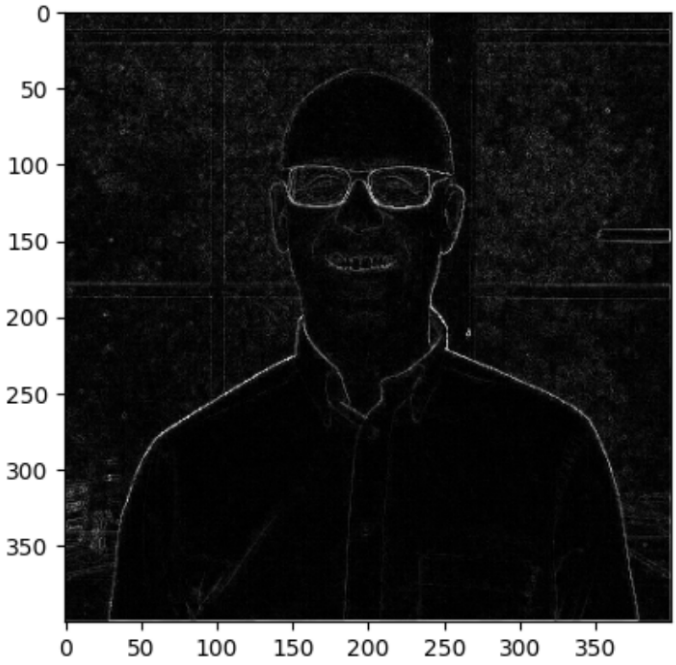
Filter 1: Prof. Frazier basically disappears, but you can still see his glasses and the edges of his shirt.



Filter 2: Prof. Frazier is looking shiny as this filter highlights the light area of the photo and emboldens all the lines

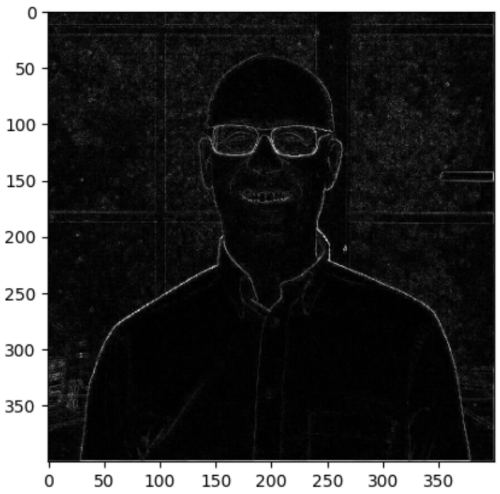


Filter 3: Here is the outline of Prof Frazier. It is amazingly detailed as you can even see the lines of his teeth!



- **Another useful method is pooling. Apply a 2x2 filter to one of your convolved images, and plot the result. In effect what have you accomplished by applying this filter?**

  The original pixel size of the picture was 800x800, and now it is 400x400.



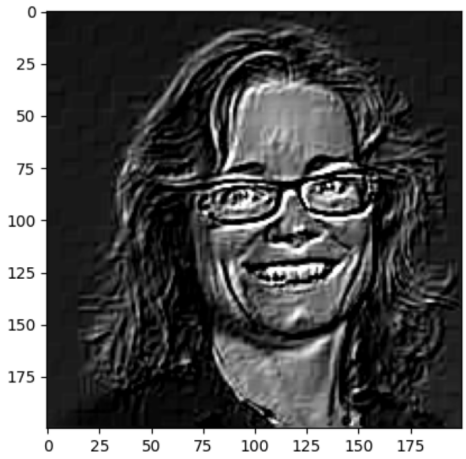
- **Does there seem to be a logic (i.e. maximizing, averaging or minimizing values?) associated with the pooling filter provided in the example exercise (convolutions & pooling)? Did the resulting image increase in size or decrease? Why would this method be useful?**

  By reducing the image pixel in a reasonable manner, we can reduce running time and simplify the running process while still keeping the essential features of the picture.

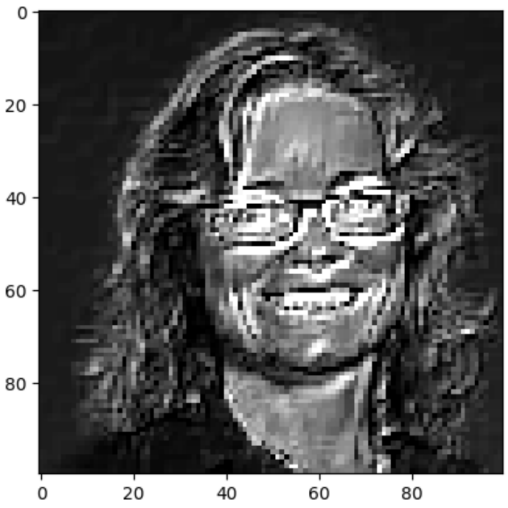
- **Stretch goal: again, instead of using misc.ascent(), apply the pooling filter to one of your transformed images.**

  This is my advisor Professor Zeman. Please don't tell her I did this, she might expel me.

  Before pooling, using filter 2. Image size is 180x180:



  After pooling, image size is 90x90:



- Convolve the 3x3 filter over the 9x9 matrix and provide the resulting matrix. link to matrices

$$\begin{pmatrix} 0 & 3 & 0 \\ 0 & 5 & 3 \\ 0 & 3 & 0 \end{pmatrix}$$